

# Red Hat Enterprise Linux 5 启动引导 Trouble Shooting 不完全实验笔记

前言：

看有些哥们很头痛这个，所以整理个笔记方便大家。

由于绝大部分内容是自己总结的，所以以下内容并不保证绝对的正确性与权威性。

版权：

GPL v3: <http://www.gnu.org/licenses/gpl.html>

Name : Yunlin Fang ( Alin Fang )

MSN: [cst05001@hotmail.com](mailto:cst05001@hotmail.com)

G Talk: [cst05001@gmail.com](mailto:cst05001@gmail.com)

27 Aug, 2008

正文：

电脑的生命过程：

这就把 jerrywjl 的帖子抄过来吧：

*Linux* 系统启动过程大致按照如下步骤进行（这是一个简述）：

第一阶段：BIOS 启动引导阶段；

在该过程中实现硬件的初始化以及查找启动介质；

从 MBR 中装载启动引导管理器（GRUB）并运行该启动引导管理

第二阶段：GRUB 启动引导阶段；

装载 stage1

装载 stage1.5

装载 stage2

读取/boot/grub.conf 文件并显示启动菜单；

装载所选的 kernel 和 initrd 文件到内存中

第三阶段：内核阶段：

运行内核启动参数；

解压 initrd 文件并挂载 initrd 文件系统，装载必须的驱动；

挂载根文件系统

第四阶段：Sys V init 初始化阶段：

启动/sbin/init 程序；

运行 rc.sysinit 脚本，设置系统环境，启动 swap 分区，检查和挂载文件系统；

读取/etc/inittab 文件，运行在/etc/rc.d/rc<#>.d 中定义的不同运行级别的服务初始

化脚本;

打开字符终端 1-6 号控制台/打开图形显示管理的 7 号控制台

**OK**, 到这里抄袭结束。

首先, BIOS 坏掉的话, 不干 Linux 的事情。

其次, 硬件坏掉, 也不干 Linux 的事情。

那么我总结了几个 Linux 在启动时可能出现的 trouble 的地方。

第一阶段:

- MBR 里面的引导管理器(IPL, Initial Program Loader)grub 损坏。

第二阶段:

- stage1.5, stage2 损坏。
- 找不到 grub.conf
- 找不到 kernel

第三阶段:

- 找不到 initrd.img
- fstab 分区挂载失败

第四阶段:

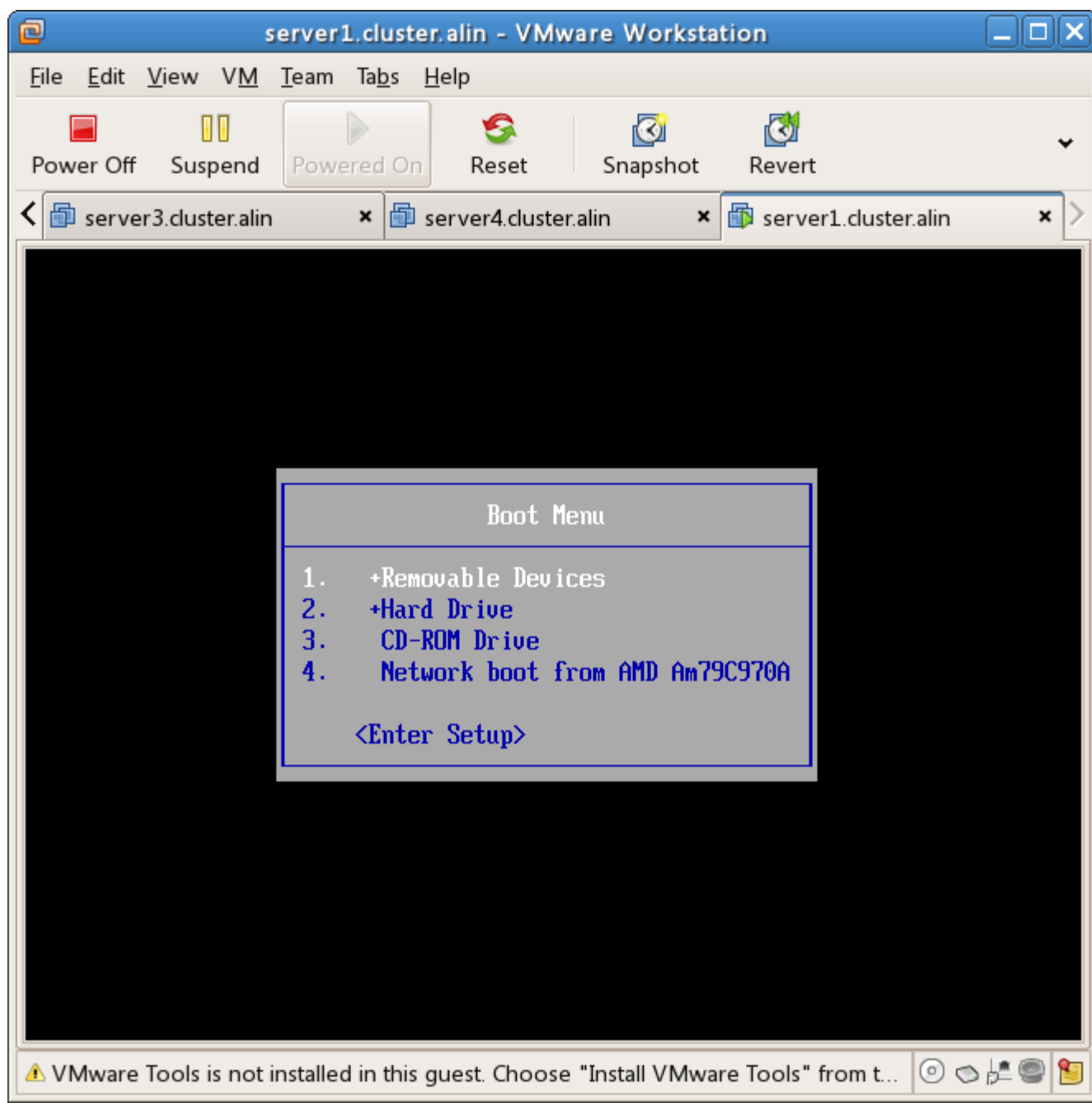
- inittab 初始化脚本失败
- 服务进程错误(内容太多, 不在讨论范围-\_-||)

## TrobuleShooting 预科班：如何进行修复模式

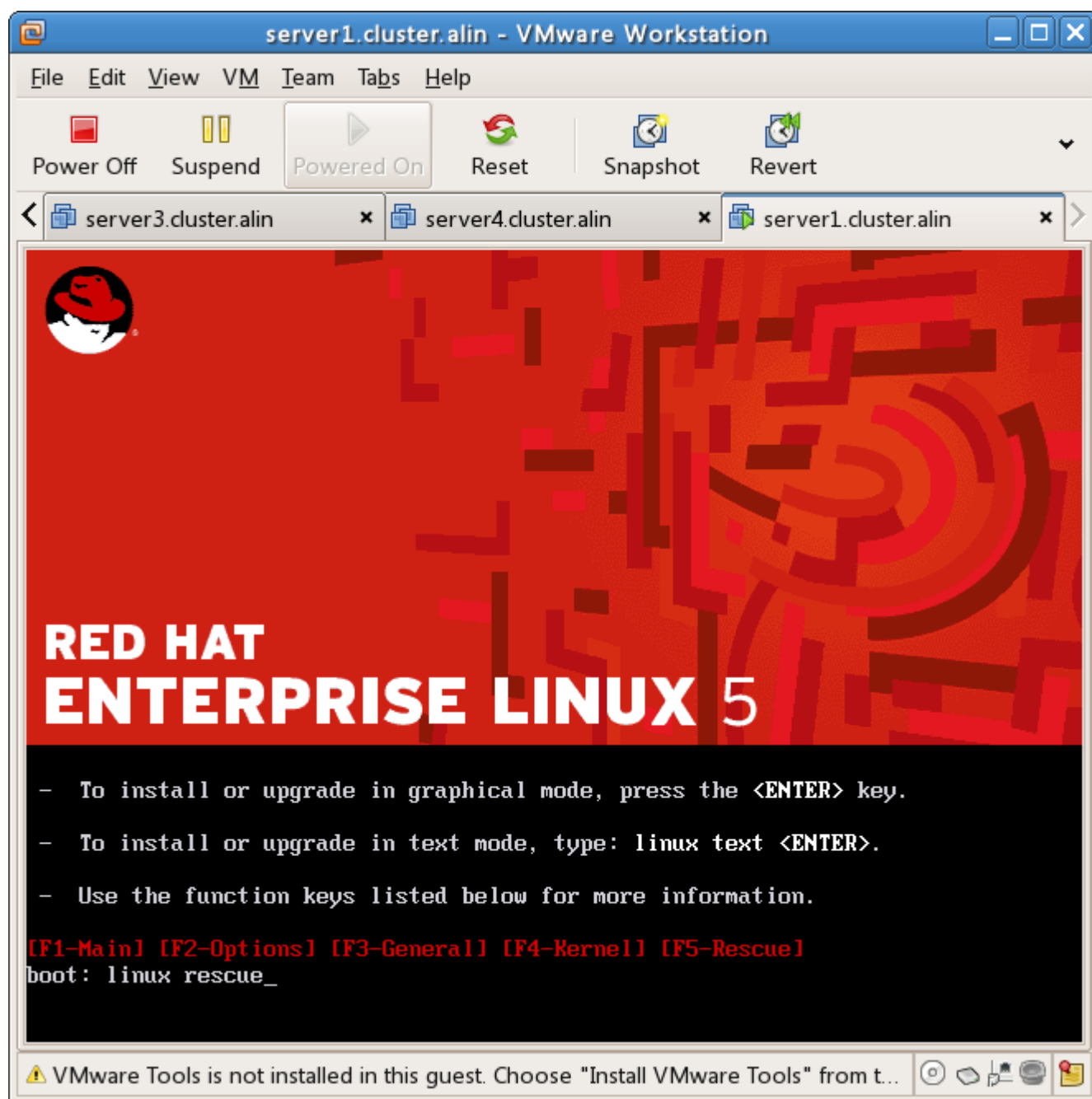
TrobuleShooting 有很多时候需要在修复模式下对系统进行修复。

这里给初学 Linux 的哥们说下进修复模式的方法。

把安装光盘放入光驱，重启电脑，选择从光驱启动。输入 `linux rescue` 回车。即可进入修复模



式。



Choose a Language: 选择语言

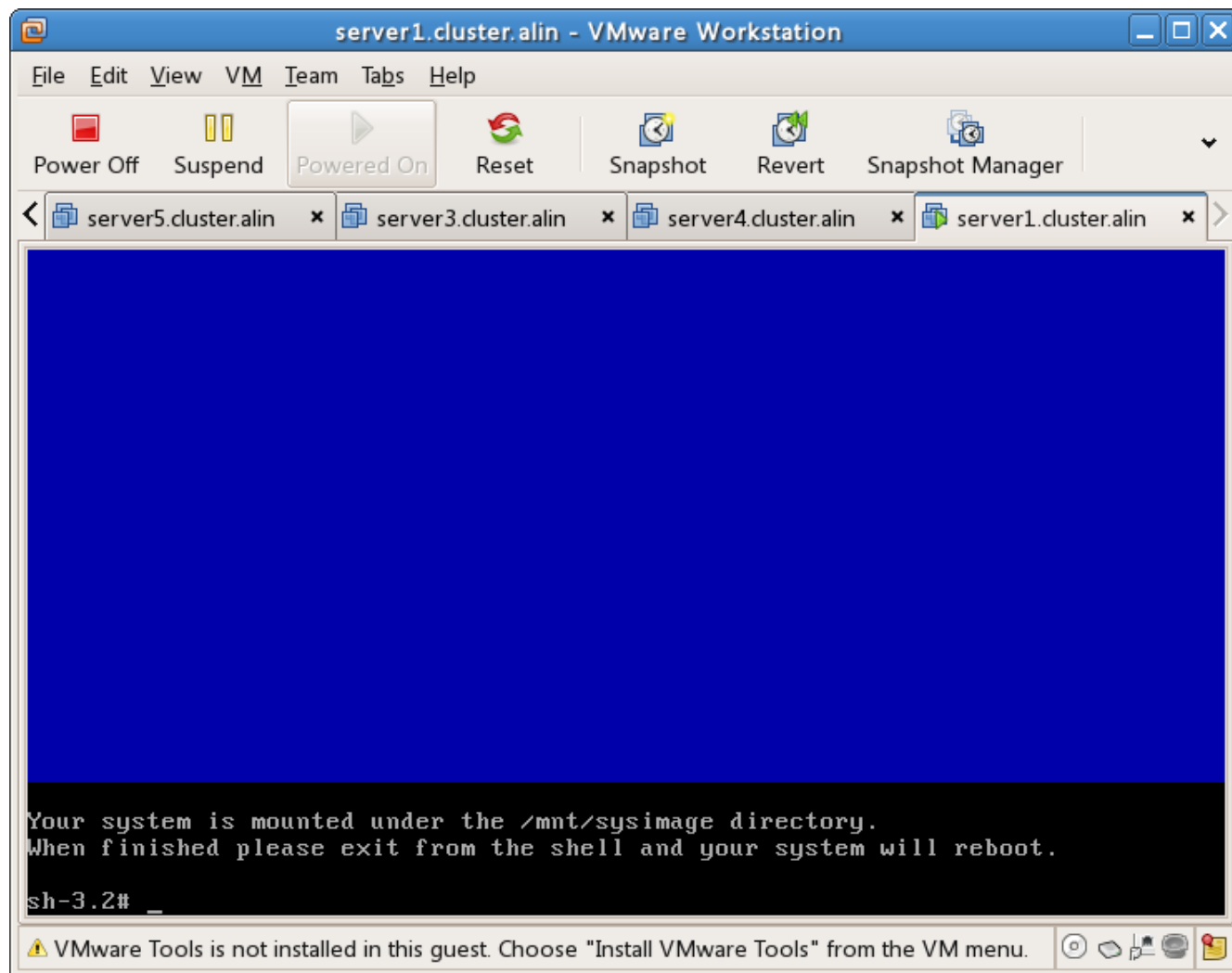
Keyboard Type: 选择键盘布局

Setup Networking: 是否启用网络支持

Rescue: 选择用何种方式挂在硬盘分区，有“继续(读写)”，“只读(有个P用.....)”，“跳过”三项。一般我们选“Continue”。

然后告诉你，修复模式下，把硬盘的/分区挂到了/mnt/sysimage 下，把/boot 挂到了/mnt/sysimage/boot 下。

这个就是进入修复模式了。



**TrobleShooting** 预科班：如何猜测硬盘的分区结构以及作用

进入修复模式。

SH-3.2# chroot /mnt/sysimage

SH-3.2# df

```
server1.cluster.alin - VMware Workstation
File Edit View VM Team Tabs Help
Power Off Suspend Powered On Reset Snapshot Revert Snapshot Manager
< Home x server3.cluster.alin x server1.cluster.alin x server2.cluster.alin x >
sh-3.2# df
Filesystem            1K-blocks      Used Available Use% Mounted on
/dev/sda3              7516200    1974748   5153484   28% /
/dev/sda1              101086      11106    84761   12% /boot
sh-3.2# fdisk -l

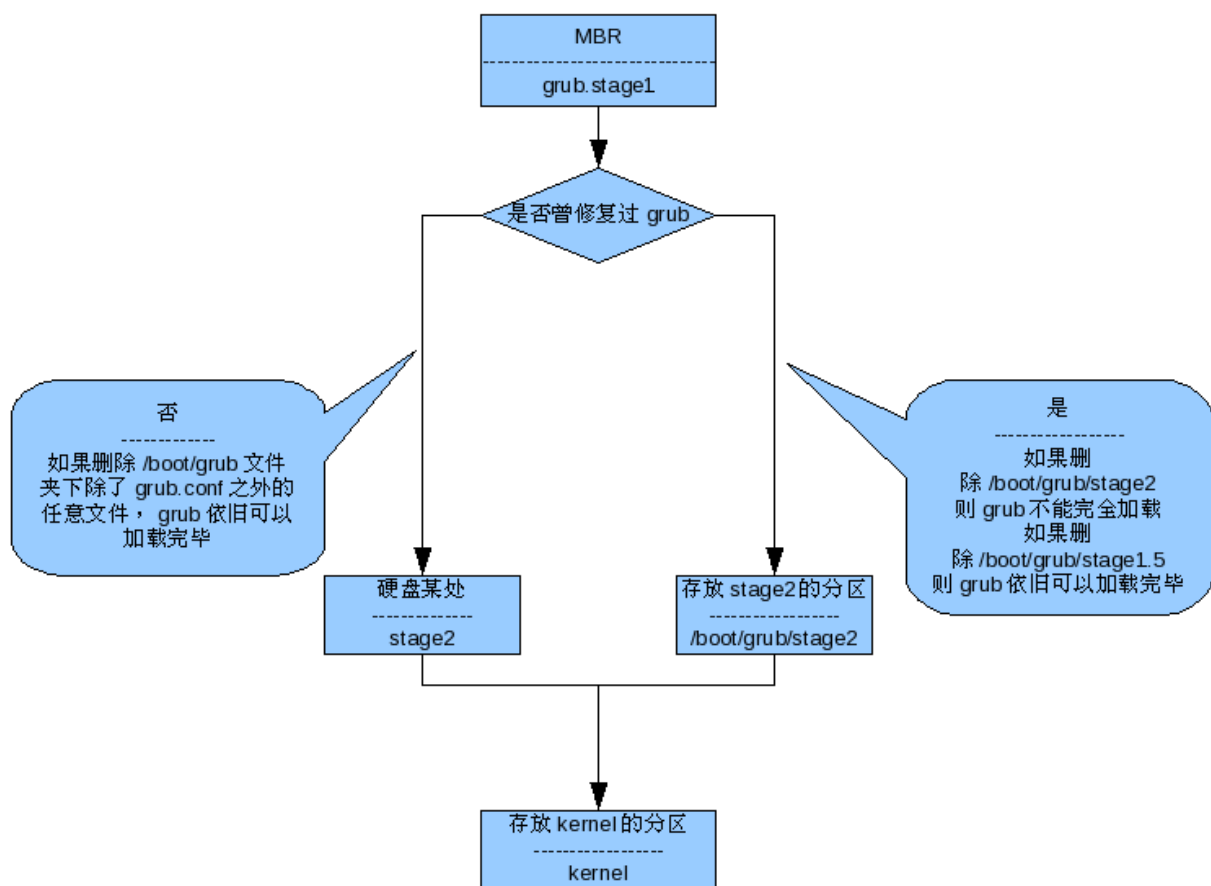
Disk /dev/sda: 8589 MB, 8589934592 bytes
255 heads, 63 sectors/track, 1044 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1  *           1           13       104391   83  Linux
/dev/sda2             14           78      522112+   82  Linux swap / Solaris
/dev/sda3             79        1044      7759395   83  Linux
sh-3.2# _

VMware Tools is not installed in this guest. Choose "Install VMware Tools" from the VM menu.
```

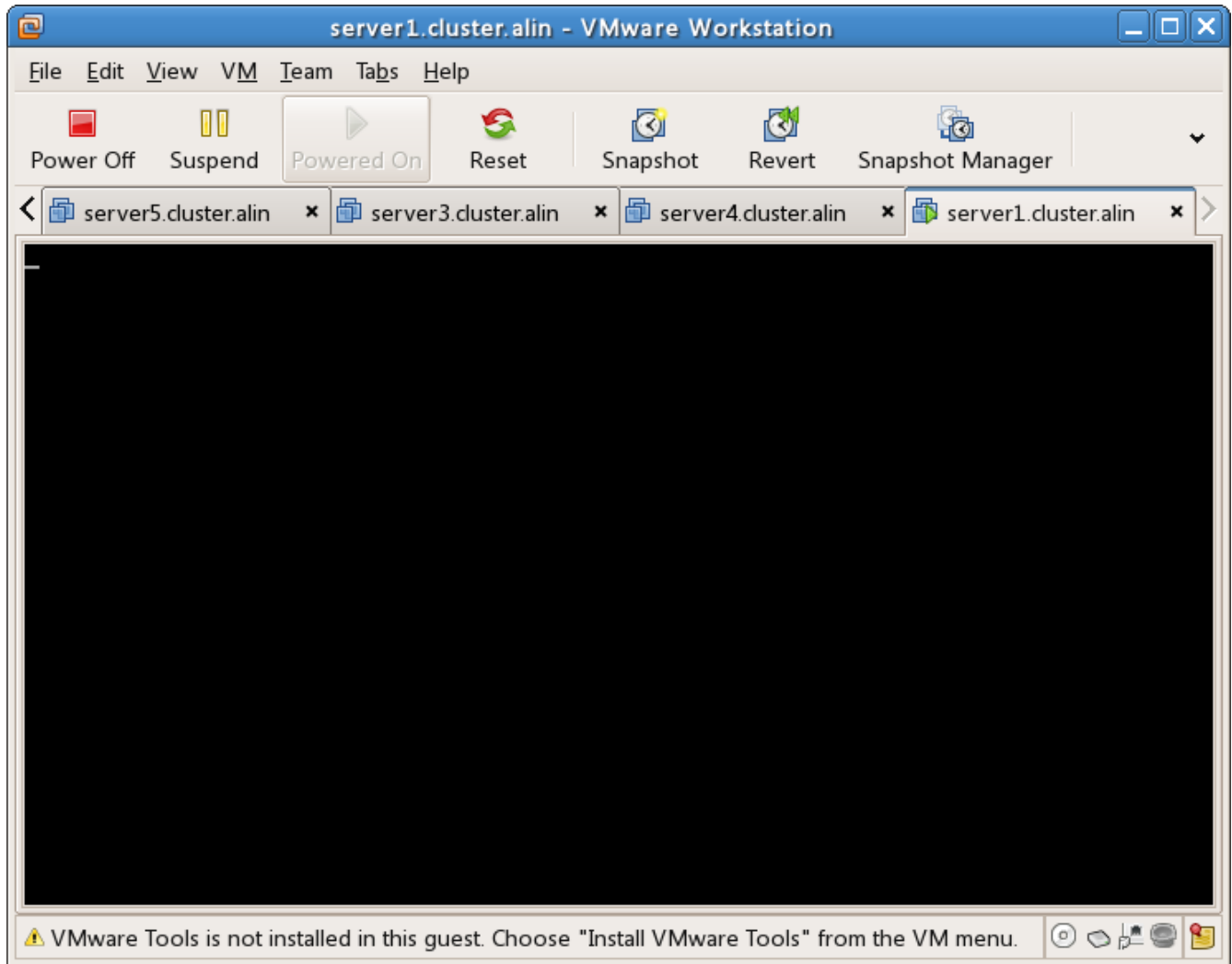
根据 df 和 fdisk 就可以推测出硬盘的分区结构以及作用了。

**TrobleShooting** 预科班： **GRUB stage1-stage2** 的过程



对症下药：

屏幕一片黑，无法看到 **grub** 界面——**MBR** 里面的 **grub** 损坏



修复：

方法 1：

进入修复模式。

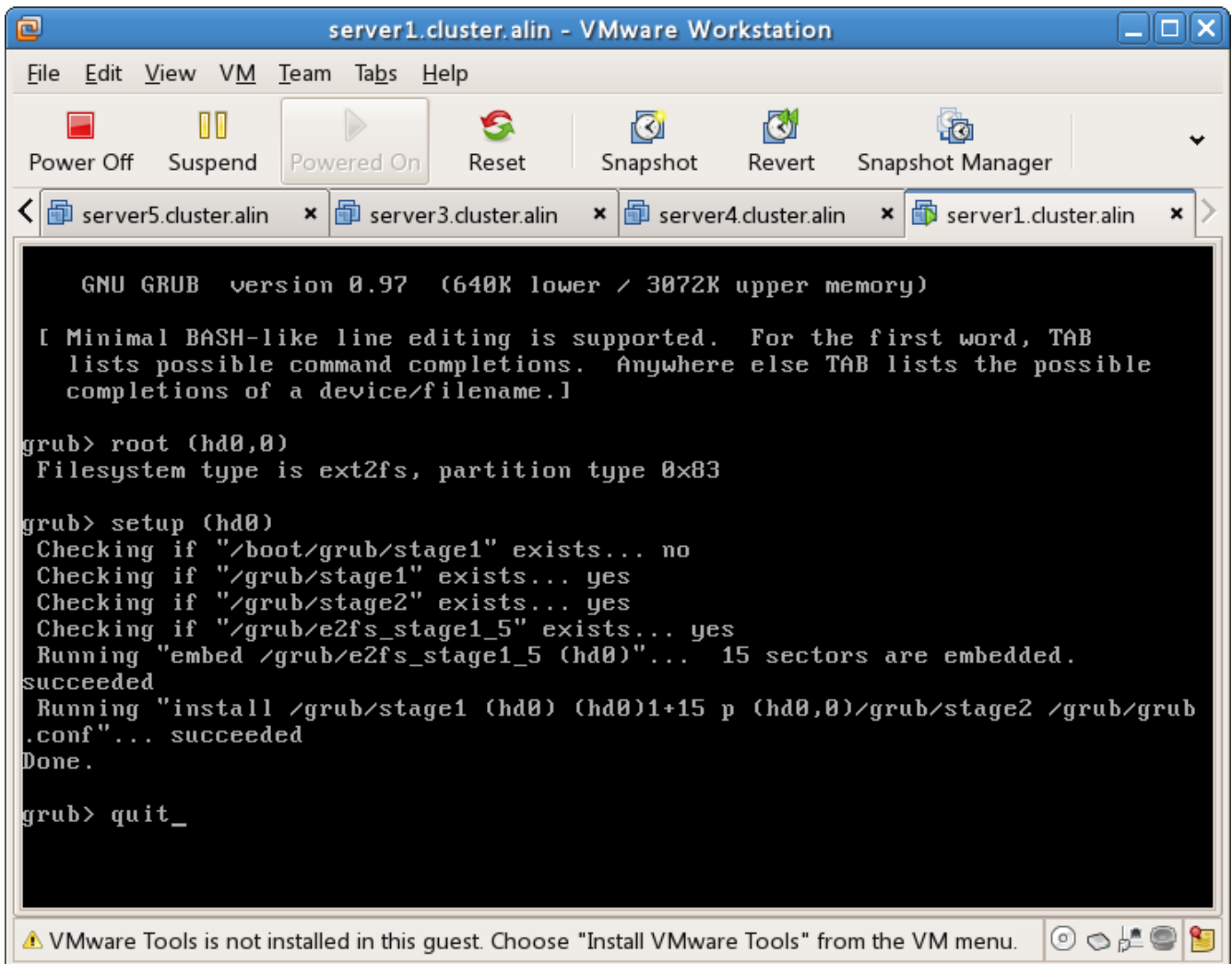
```
SH-3.2# chroot /mnt/sysimage
```

```
#> grub
```

```
grub> root hd(0,0)
```

```
grub> setup (hd0)
```





如果修复成功，grub 会提示你 success 了。

注意：

- 此方法修复 grub 依赖于/boot/grub/grub.conf，如果 grub.conf 不存在，则需手工创建一个。创建方法请参考下文。
- 用此方法修复 grub 是把/boot/grub 下的 stage 镜像安装到硬盘上的特定位置上。所以使用此法修复 grub 仅限于硬盘上的 stage1(在 MBR 里)以及 stage1.5，修复依赖于/boot/grub 下的 stage1, stage1.5, stage2 镜像。如果这几个镜像也遭到破坏，则此法无效。

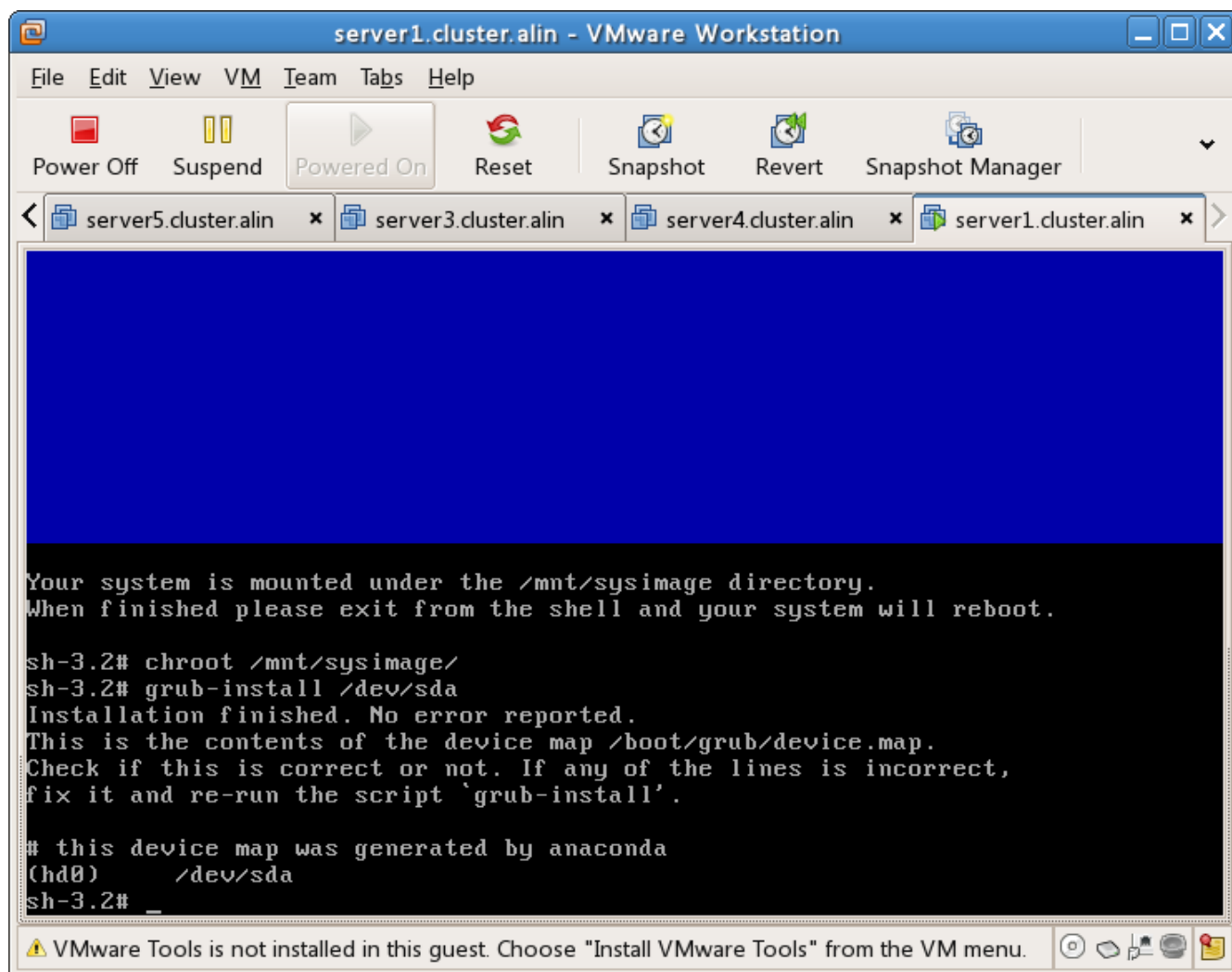
方法 2:

进入修复模式。

```
SH-3.2# chroot /mnt/sysimage
```

```
SH-3.2# grub-install /dev/sda
```

如果似乎成功，会如此提示你：



注意：

- 此方法修复 grub 依赖于 /boot/grub/grub.conf，如果 grub.conf 不存在，则需手工创建一个。创建方法请参考下文。
- 此方法不依赖于 grub.conf，当 /boot/grub 下文件都找到破坏，亦可用次命令修复。但不修复 grub.conf。

**grub 警告： Error 15: File not found——找不到 *kernel* 或者 *initrd.img***

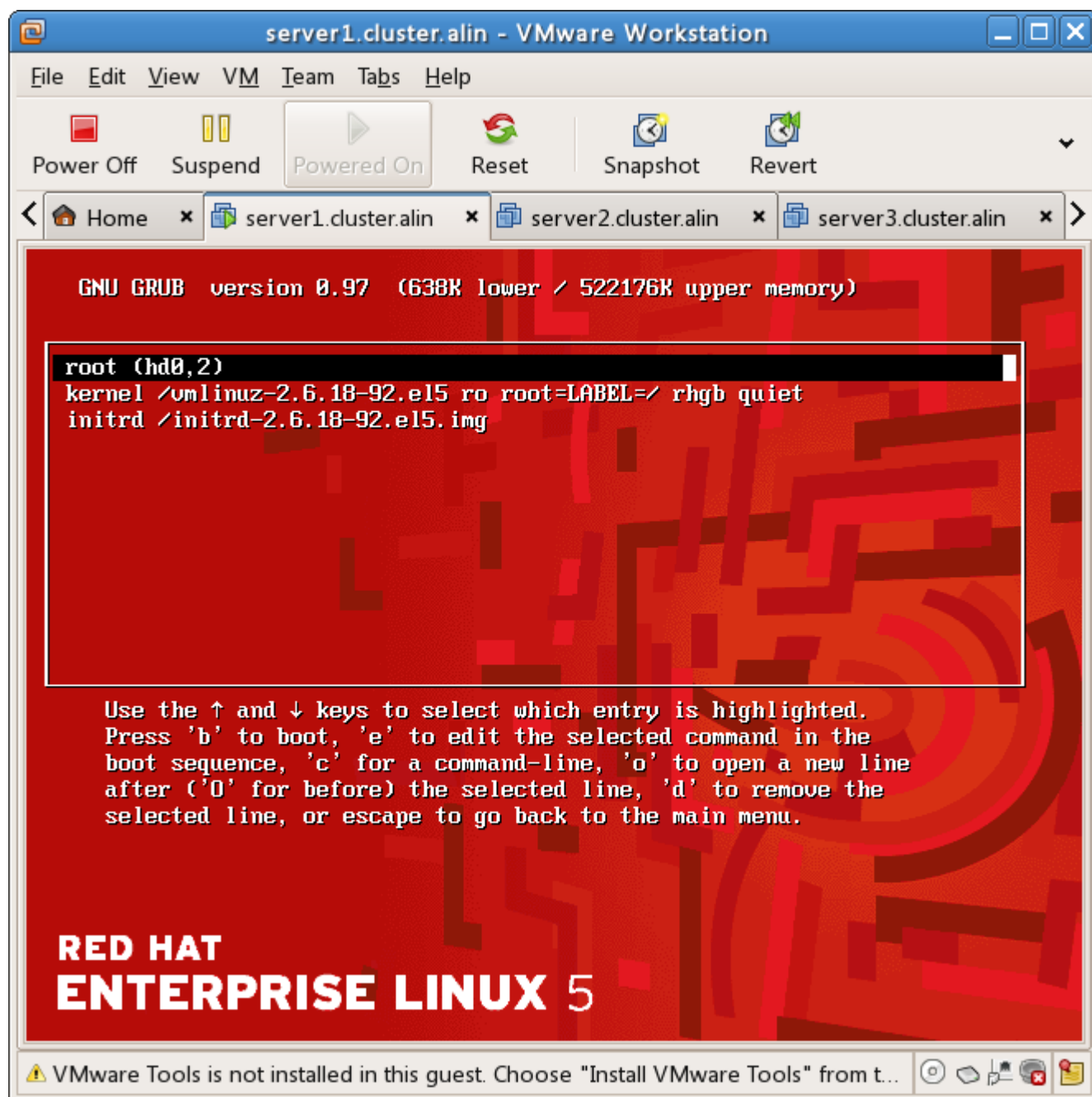
Stage1.5 报错。

修复：

路线 1：

在 grub 界面，在要加载的内核选项处按<e>进入编辑模式，再在带检查的各栏按<e>，利用<TAB>补全功能，

- 确定 kernel 以及 initrd.img 文件是否存在,
- 确定指定 kernel 所在磁盘分区的 root 参数是否写对,



就拿上面截图来说，最容易出问题的地方有：

第一行 root (hd0,0)

这里的 root 指定的是内核所在分区，不是根文件系统所在的分区。

这里不用/dev/sdxn 这样的格式来标识硬盘分区。而是直接用 hd(n,m)这样的格式来标识硬盘。

N 表示第几块硬盘,m 表示是第几个分区。这里 m 不管是逻辑分区还是主分区。全部按照从 0 向后的顺序递增。

比如:

hd(0,2)表示第一块硬盘上的第三个分区。

hd(1)表示第二块硬盘。

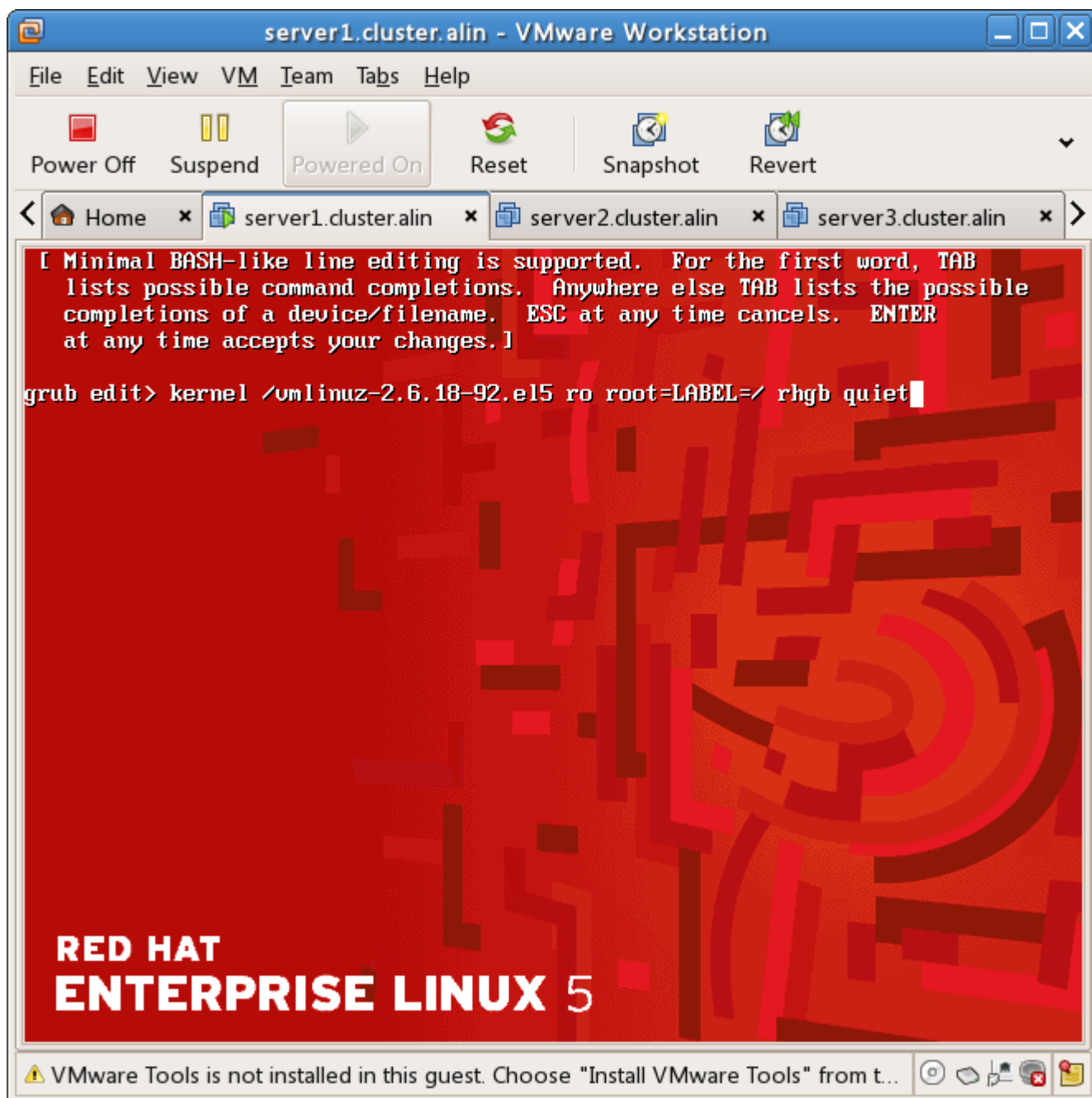
第二行 **kernel /vmlinuz** 的位置。

一个是要检查内核位置是否是以/boot 所在分区为根的绝对路径。比如如果你是单独把/boot 划出来当作一个分区，那么如果你如果内核在以 linux 的根分区为准的绝对路径是/boot/vmlinuz 的话，那么你以/boot 分区 为根，则要写/vmlinuz

initrd.img 亦如上。

一个是要确定

且判断导致 **Error 15** 错误是因为 grub.conf 里的配置内核模块的参数是否写错，尤其是文件名写错，则改正之。并按<b>或者再次进入安全模式，进入可用系统后修改/boot/grub/grub.conf 参数。



利用<TAB>自动补全功能检查/boot/grub/grub.conf 配置是否正确。

如果发现内核丢失，则走路线 2。

如果发现 initrd.img 镜像丢失，则走路线 2 或者路线 3。

路线 2:

进入修复模式，

挂载安装盘，强制重装内核。

```
SH-3.2# chroot /mnt/sysimage
```

```
SH-3.2# mount -o loop -t iso9660 /dev/sdc /media
```

```
SH-3.2# cd /media/Server
```

```
SH-3.2# rpm -ivh --force kernel-2.6.18-92.el5.i686.rpm
```

```
SH-3.2# reboot
```

路线 3:

进入修复模式，重新生成镜像文件。

```
SH-3.2# chroot /mnt/sysimage
```

```
SH-3.2# mkinitrd initrd- 2.6.18-92.el5.img `uname -r`
```

```
SH-3.2# reboot
```

为什么走路线 2 强装内核也可以呢？因为在 RHEL5 里面安装 kernel 的 RPM 包，安装脚本会帮你执行 mkinitrd 这个步骤。

**Grub>** 直接进入 **grub shell** 界面——找不到 **grub.conf** 文件

重装 grub？

No。

有些配置文件是 rpm 自带的。在安装 rpm 时会自动生成。

但是，有些配置文件是根据你机器配置事实生成的。这种配置文件就不一定能靠重装相关的包来修复。

怎么办？自己手动重写。

修复：

进入修复模式。

```
SH-3.2# chroot /mnt/sysimage
```

```
SH-3.2# echo /boot/vmlinuz-2.6.18-92.el5 >> /boot/grub/grub.conf
```

```
SH-3.2# echo /boot/initrd-2.6.18-92.img >> /boot/grub/grub.conf
```

```
SH-3.2# vim /boot/grub/grub.conf
```

```
title Red Hat Enterprise Linux 5 update 2
```

```
root (hd0,0)
```

```
kernel /vmlinuz-2.6.18-92.el5 ro root=/dev/sda3
```

```
initrd /initrd-2.6.18-92.img
```

重启。

**grub** 警告: **Error 15——stage1.5** 报错, 极有可能是 **stage2**, **stage1.5** 加载 **stage2** 失败。

Stage1.5 报错。

修复:

进入修复模式。

Grub-install ?

不一定能成功。

grub-install 的安装必须依赖 grub.conf 这个配置。如果这个配置不存在, grub 是 install 不了的。

所以你要现确认/boot/grub 下的 grub.conf 是否存在。

如果不存在, 参照上面修复 grub.conf 的做法。

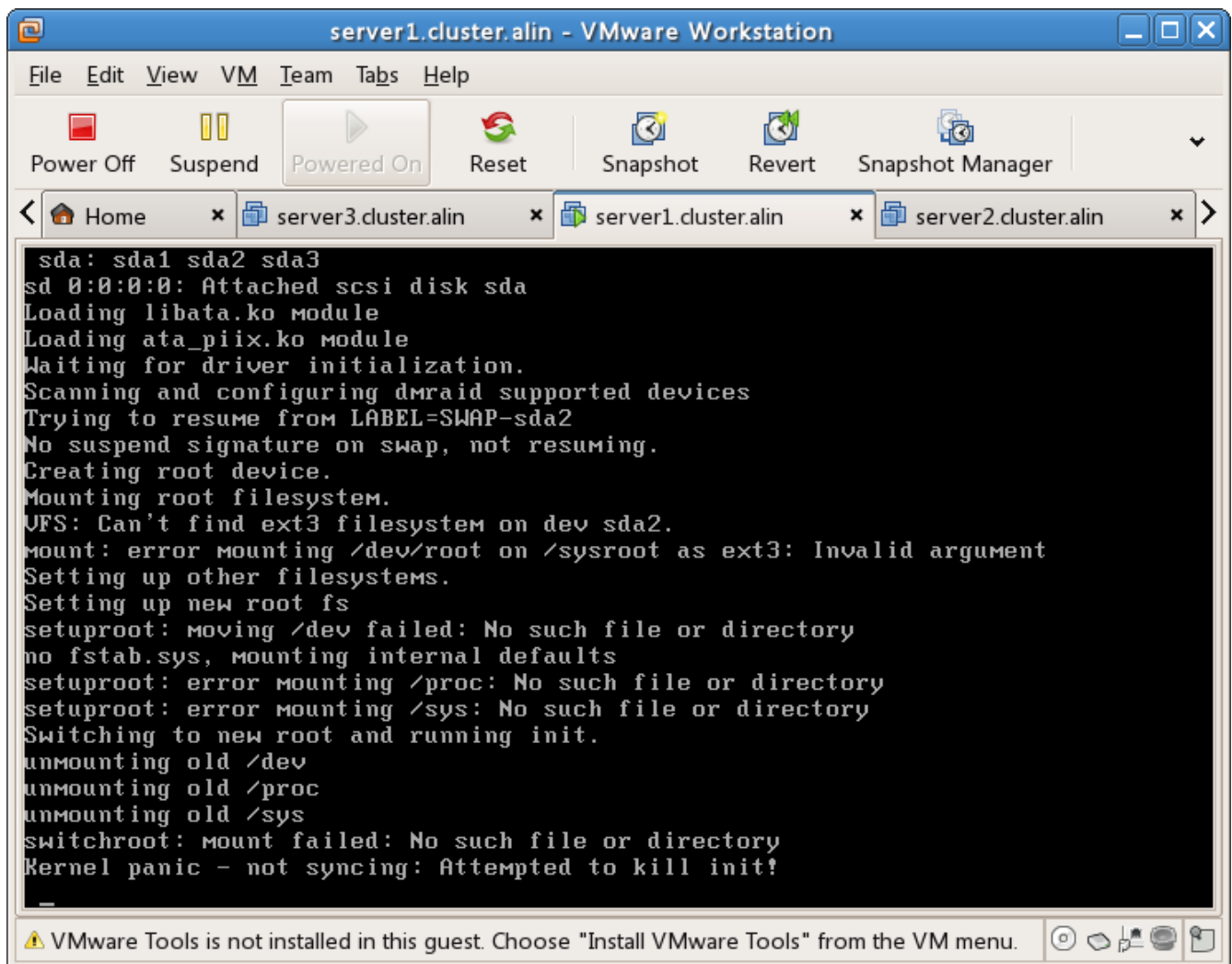
修复 grub 之后, 执行重装 grub 即可。请参见之前修复 grub 的方法。

注意:

- 初始状态的 **stage1** 并不会去读取/boot/grub/下的 **stage1.5** 和 **stage2**。位于 MBR 的 **stage1** 会去读取硬盘某个位置的 **stage1.5**, 然后直接引导内核。
- 如果机器上的 **grub** 是经过修复的, 比如用 **grub-install** 或者进入 **grub** 然后 **setup** 过。那么就不一样了。此时 **stage** 会去引导/boot/grub 下的 **stage2**。

**Kernel panic——**原因有 **N** 多。

Kernel panic 的原因太多了。主要是你要看是在系统运行的哪个阶段 kernel 喊痛了。



比如说这个阶段。

### Switchroot: mount failed: No such file or directory

Kernel panic – not syncing: Attempted to kill init!

内核挂 root 载分区失败。

常见的原因有 root 分区遭到破坏，或者内核不能正确定位 root 分区。

如果是 root 分区遭到破坏，那么这个不在我们讨论的范围之内。

如果是内核不能正确定位 root 分区，则我们可以尝试修复之。

通常情况下内核如何定位 root 分区呢？一般是通过 grub 在引导起内核的时候，给内核指定 root 分区的位置。

大家看这个 grub 配置：

default=0

timeout=5

splashimage=(hd0,4)/grub/splash.xpm.gz



hiddenmenu

password --md5 \$1\$YfEwIjnz\$cCLdMBF3FDn1iww9fSMp/.

title Red Hat Enterprise Linux Client (2.6.18-92.el5)

root (hd0,4)

kernel /vmlinuz-2.6.18-92.el5 ro **root=LABEL=**/ rhgb quiet

initrd /initrd-2.6.18-92.el5.img

一般情况下，grub 引导一个 linux 内核，有这几行参数：

- title
  - title 指定了要引导的内核的 title。顾名思义。
- root
  - 这个 root 并不是根分区的意思。这里是指定了内核所在的文件系统的位置。此 root 非彼 root。
- Kernel
  - 这里指定了要加载的内核以及加载内核的参数。
- initrd
  - 这里指定了要加载的 initrd.img 模块。

从这个思路走，我们检查 grub.conf。

修复：

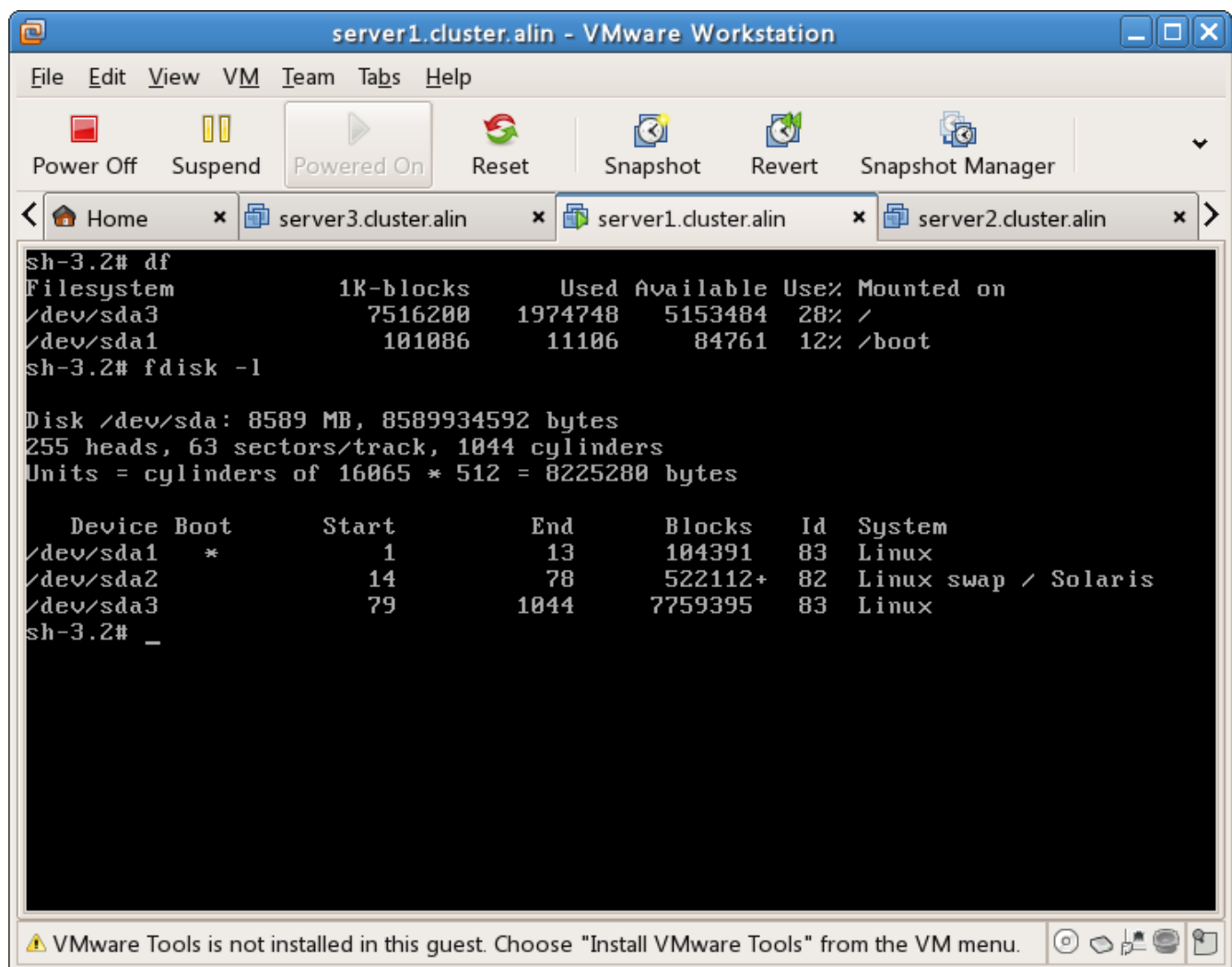
进入修复模式。

SH-3.2# chroot /mnt/sysimage

SH-3.2# df -h

SH-3.2# fdisk -l

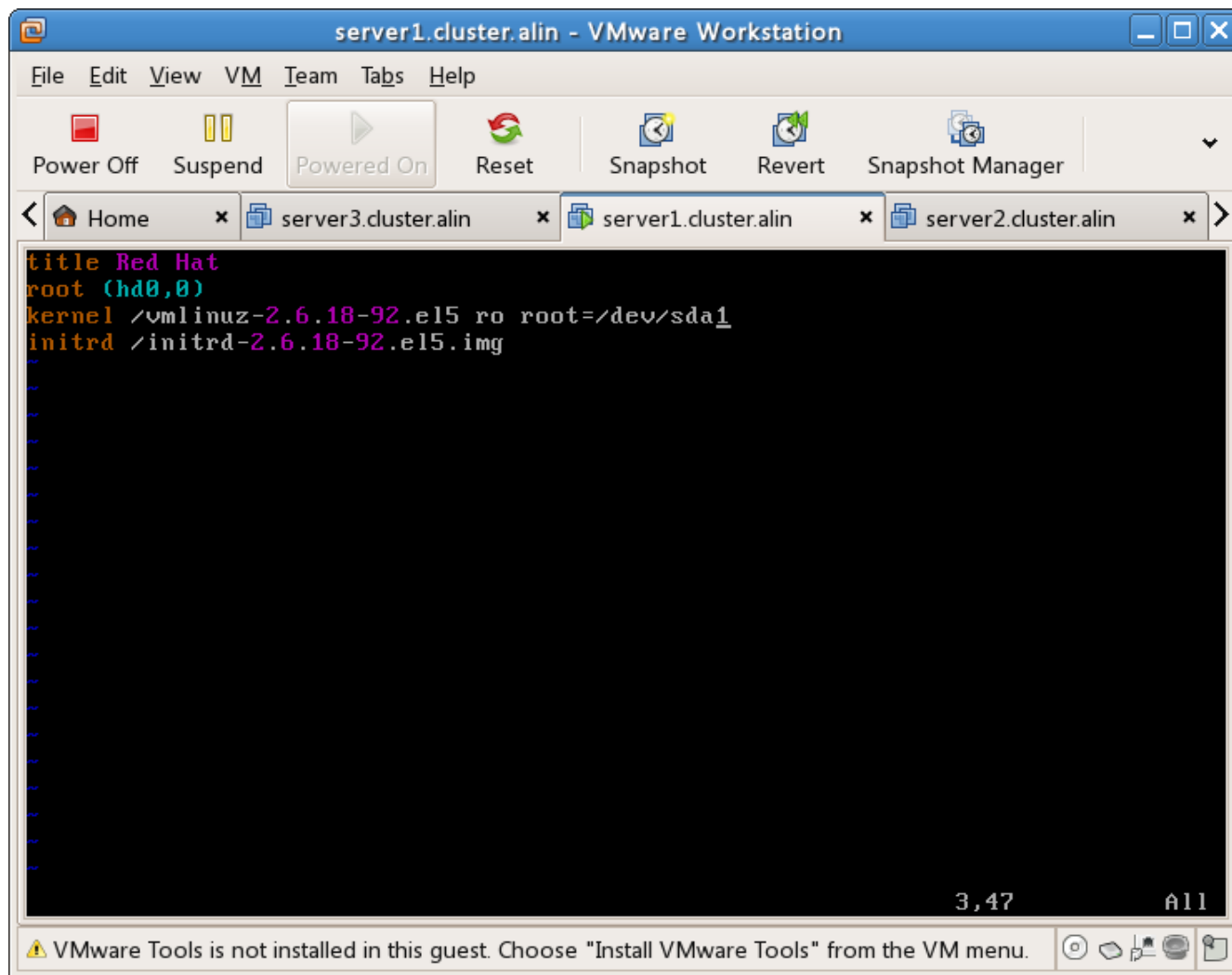
确定根分区在硬盘上的位置。



SH-3.2# vim /boot/grub/grub/conf

检查 kernel 选项参数。

核对 root 参数是否正确。



这里发现本应指向 linux 根分区的 kernel 选项的 root 参数指向了 /boot 分区，kernel 在 /boot 分区找不到/目录该有的东西，所以就 kernel panic 了。

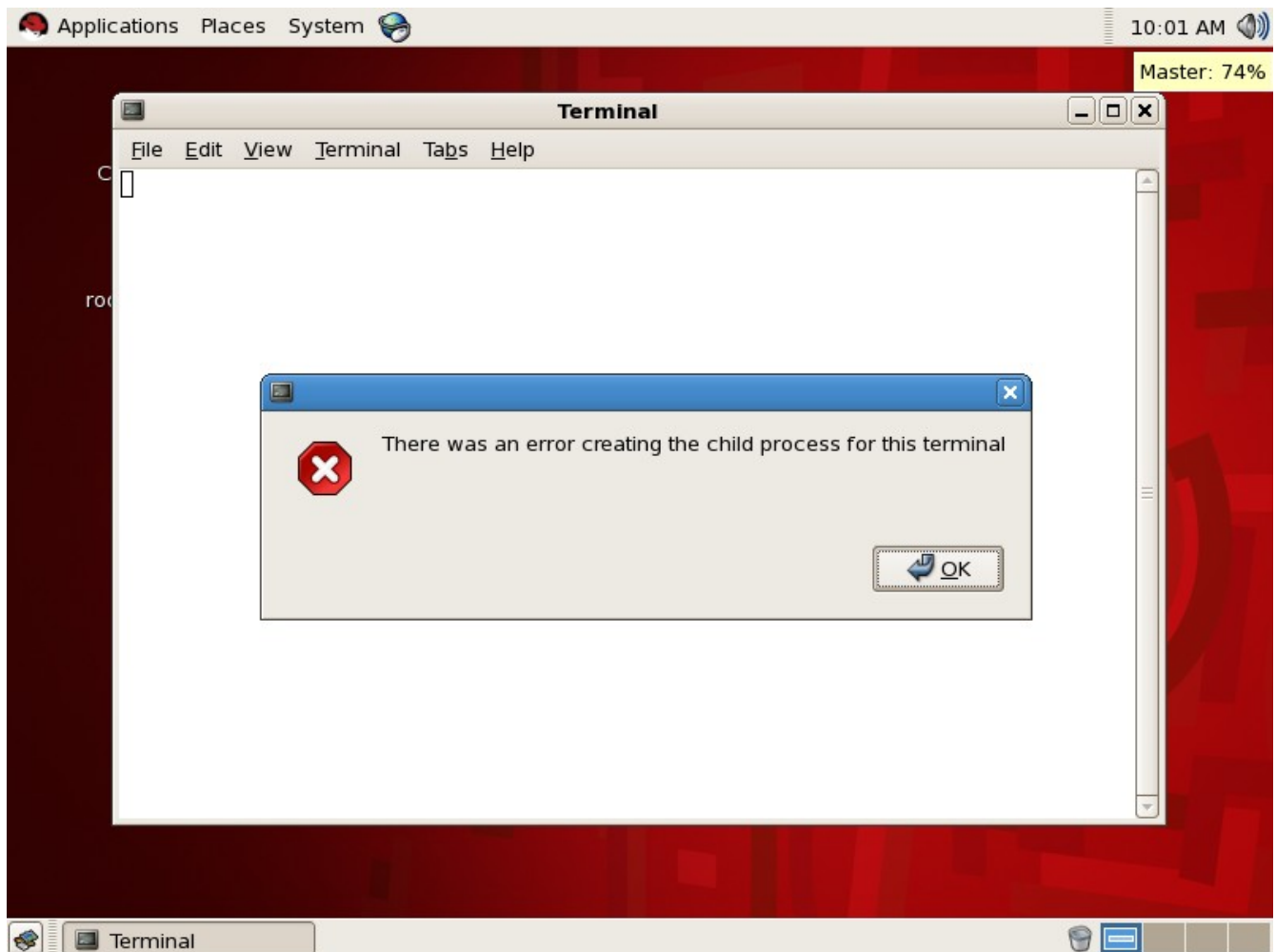
改正。

重启。

OK，到此说的 **trouble** 都是在 **linux** 内核完全加载完毕之前。现在开始将 **linux** 内核加载之后的 **trouble**。

**X** 下打开虚拟终端出现警告：**There was an error creating the child process for the terminal**

**ssh** 到此机器，连接成功，输入用户名，输入密码，并无提示密码错误，并不断开连接，终端无反应——原因：由于作者水平有限，很难解释清楚。只知道已有可能的诱因。



RHEL5 里面，虚拟终端 pts 是依赖于设备 devpts 的。如果 devpts 未挂载，则 pts 虚拟终端是起不来的。

修复：

检查 devpts 是否被挂载

```
[root@server1 ~]# mount
```

```
/dev/sda3 on / type ext3 (rw)
```

```
proc on /proc type proc (rw)
```

```
sysfs on /sys type sysfs (rw)
/dev/sda1 on /boot type ext3 (rw)
tmpfs on /dev/shm type tmpfs (rw)
none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw)
/dev/hdc on /misc/cd type iso9660 (ro,nosuid,nodev)
```

发现 devpts 没有给挂载

```
[root@server1 ~]# vim /etc/fstab
```

编辑挂载选项:

```
LABEL=/          /          ext3 defaults    1 1
LABEL=/boot      /boot      ext3 defaults    1 2
tmpfs            /dev/shm   tmpfs defaults    0 0
devpts           /dev/pts   devpts gid=5,mode=620 0 0
sysfs            /sys       sysfs defaults    0 0
proc             /proc      proc  defaults    0 0
LABEL=SWAP-sda2  swap       swap  defaults    0 0
~
```

gid=5 是什么意思?

```
[root@server1 ~]# head -n 6 /etc/group
```

```
root:x:0:root
```

```
bin:x:1:root,bin,daemon
```

```
daemon:x:2:root,bin,daemon
```

```
sys:x:3:root,bin,adm
```

```
adm:x:4:root,adm,daemon
```

```
tty:x:5:
```

```
[root@server1 ~]#
```

明白了吧?

=====

OK, 到此位置。由于种种原因, 更多的加载 kernel 之后的 Trouble Shooting, 我就不多说了。但是要记住, 由于这个时候 Linux 已经开始启动, 关联的东西更加错综复杂, 从此开始的 Trouble Shooting 也更加的凶险与有趣。